Isaac EtherNet/IP™ Guide

Abstract

This document is to be used by controls engineers and other technical personal responsible for communications between programmable logic controllers (PLC's) and the Isaac multi-tester by Zaxis Inc. The document assumes the reader is well acquainted with EtherNet/IP™ as defined the ODVA association. The reader should also have a good working knowledge of TCP/IP and UDP/IP protocols.

Last updated
12-Aug-19

Table of Contents

Isaac EtherNet/IP™ Guide 0
Introduction 1
EtherNet/IP™ for the Isaac tester overview 2
Isaac Tester Configuration for Ethernet/IP™ 3
PLC Configuration for Ethernet/IP™ 3
Configuration with EDS file 4
Generic Ethernet Module configuration 7
Isaac Tag Groups 10
Configuration Tags 10
cosType – Change-Of_State flags 11
PLC Output Tags 11
PLC Input Tags 12
COS Change-of-State bits 14
Isaac Firmware/Program tags 14
Firmware Parameters 14
Reading and Writing of Firmware Parameters 16
Program Parameters 19
TCP/IP Network Configuration 23
Trouble Shooting 26

Introduction

Zaxis's Isaac multi-tester is a high-performance air leak tester with unparalleled test stability and repeatability. Depending on the model, the following tests can be performed: pressure decay, vacuum decay, mass flow, pressure cracking, burst pressure, valve cracking pressure and more. From one channel iKits to our four channel Isaac HD models EtherNet I/P is an optional communications method.

Communications with the Isaac can be accomplished in different ways, RS232/485, TCP/IP and EtherNet/IP™. This document will describe how to setup EtherNet/IP™ using an Allen-Bradley 1769 CompactLogix Controller. Other types of PLCs can be used if compliant to the EtherNet/IP™ specifications as defined the ODVA association.

EtherNet/IP™ for the Isaac tester overview

The "IP" in "EtherNet/IP" refers to "Industrial Protocol" and should not be confused with Ethernet or the TCP/IP and UDP/IP protocols. Although EtherNet/IP™ uses these technologies and protocols to accomplish the desired communications between the Isaac and PLC's.

The Isaac software consists of a number of different modules, as for EtherNet/IP™ the software follows the recommended implementation as defined in the ODVA specifications. See *The CIP Networks LIBRARY Volume 1 and 2.* The Isaac EtherNet/IP™ software levels are: Application, Session, Transport, Network, Data Link, Physical as show below in Table 1.

Application	Isaac EtherNet/IP
Session	Explicit/Implicit Messaging (CIP)
Transport	TCP/UDP
Network	Internet Protocol (IP)
Data Link	Ethernet
Physical	Peer-to-peer, multicast, unicast

Table 1 - Isaac EtherNet/IP™ Adaptation of CIP

The Application and Session layers (as shown in top two rows) are customized for the Isaac, while the lower layers (as shown in bottom four rows) follow the IEEE 802.3 standards with little or no customization. Thus, allowing existing Ethernet switches and/or networks to communicate with the Isaac.

The Isaac supports both DHCP and static Ethernet configurations. Factory default use the following static network configuration:

I/P Address	192.168.2.130
Network Mask	255.255.255.0
Gateway	192.168.2.1

Table 2 - Factory Default Network Configuration

These defaults can be changed using the TSi (Touch Screen interface) or PLC MSG command as described below. It is recommended that a static configuration be used to ensure devices that communicate to the Isaac can rely on the same I/P addresses for each Isaac tester. However, it is possible to use DHCP if the DHCP server and/or Name server is able to resolve an Isaac tester hostname to a static I/P address.

Because EtherNet/IP™ relays on TCP/UDP it is necessary that the Isaac tester and any device wishing to communicate with the Isaac tester over Ethernet have the correct network configuration. The network *ping* command can be used to diagnose network problems etc.

EtherNet/IP™ uses two forms of messaging:

- **Unconnected messaging** is used to: 1) establish connections, 2) low-priority messages, and 3) infrequent messages. Unconnected messages are handled by the Unconnected Message Manager, e.g. the UCMM software module.
- **Connected messaging** allocates resources that are dedicated to a particular purpose, such as real-time I/O data transfers. Connection resources are reserved and configured by the UCMM (Unconnected Message Manager) and then allocated to the desired resource.

Connections are created by a Connection Originator (or Originator for short) and the device that responds to the connection request, called the Connection Target (or just Target).

EtherNet/IP™ specifics two types of message connections:

- **Explicit message connections** are point-to-point relationships. These would be considered general purpose connections and are used to access general information about the Isaac tester. E.g. revision number, serial number, etc. Explicit messages almost always use TCP/IP connections to communicate.
- **Implicit (I/O data) connections** are created to communicate application-specific I/O data at regular intervals. Implicit messages use UDP/IP resources to communicate.

The Isaac tester uses Explicit messages to create a connection with the other devices (e.g. a PLC's etc.) and when requested creates an implicit connection to periodically return the current status of the Isaac tester. E.g. busy, sequence/state, current program etc.

For further information and documentation on message types etc. see OVDA publication PUB00138_EtherNetIP_Ethernet_Technology.pdf.

Isaac Tester Configuration for Ethernet/IP™

Other than configuring the correct Ethernet (TCP/IP) parameters on the Isaac, no other configuration is required for Ethernet/IP™ communications. Taking note of the Isaac tester's I/P Address, netmask and gateway, (see Table 2 - Factory Default Network Configuration) are required to configure the devices that communicate with the Isaac tester.

PLC Configuration for Ethernet/IP™

There are two ways to configure an Allen Bradly PLC to communicate with an Isaac tester.

1. Download the EDS (Electronic Data Sheet) file from the Zaxis, Inc. web site https://www.zaxisinc.com/downloads/. Then follow the steps described below in the Configuration with EDS file section below.
2. Add a generic EtherNet I/P module to your project and configure the module using the steps described below. See the Generic Ethernet Module configuration below.

Configuration with EDS file

Once the EDS file, and any other accompanying Logix user defined types have been downloaded from the Zaxis, Inc. web site the EDS file should be registered using the Rockwell Automation's EDS Wizard. A number of resources can be found on the Internet showing the proper procedure to register an EDS file.

Once the Isaac tester EDS file has been registered the configuration can be done as follows:

1. Start Logix Designer and Ensure the Communications are offline.
2. Create a new Isaac Ethernet module by right clicking on the **Ethernet** menu item displayed on the **Controller Organizer** window (Figure 5) and select **New Module…** which should display the **Select Module Type** window (Figure 6).

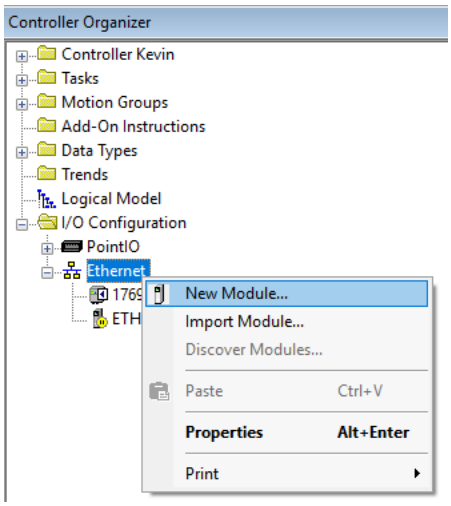


Figure 1 – Create New Ethernet-Module

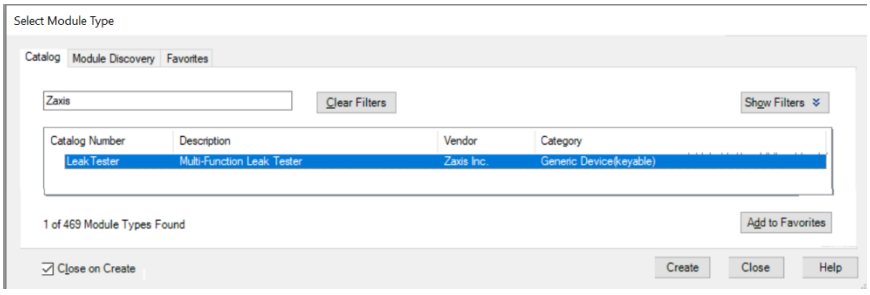
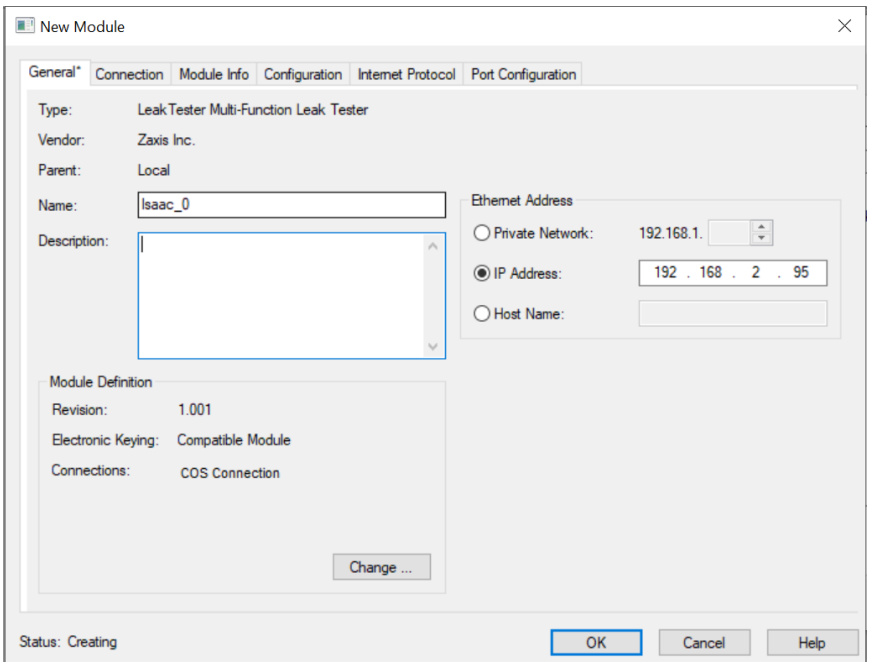


Figure 2 - Select Zaxis Leak Tester

3. Click the Create button to create the Leak Tester module, which should display the following window.



4. Enter a name for the Isaac tester and enter the assigned I/P address for the tester. A unique I/P address is often assigned to the Isaac test when ordered and will be displayed on the tester. If not use the default I/P address as given in Table 2.
5. Clicking on the **Change …** button will allow changing the type of connection from COS (change-of-state) to I/O. However, unless suggested by a Zaxis support personal, it is recommended that the COS connection be used.

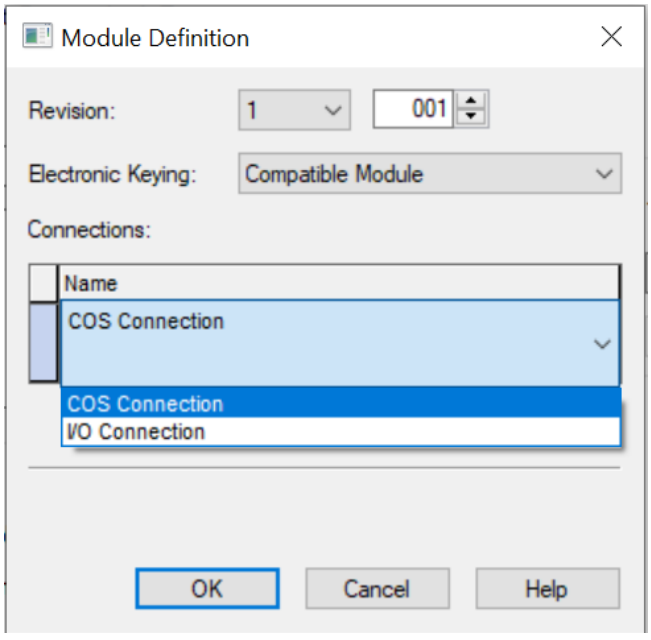
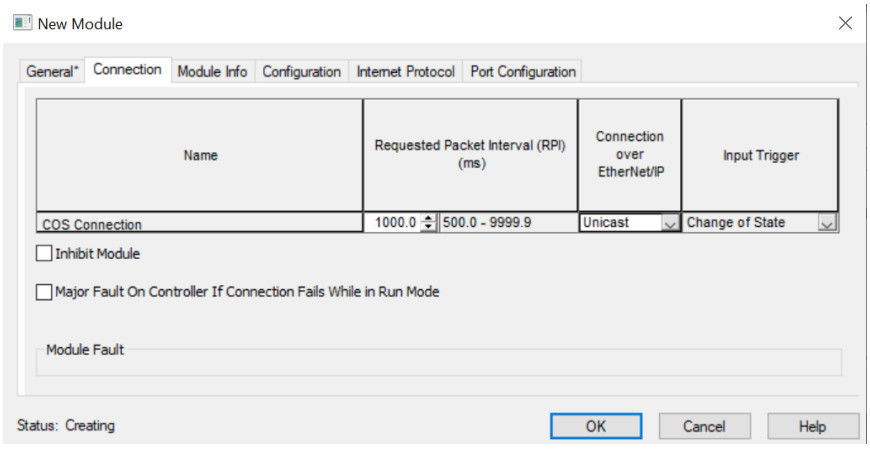
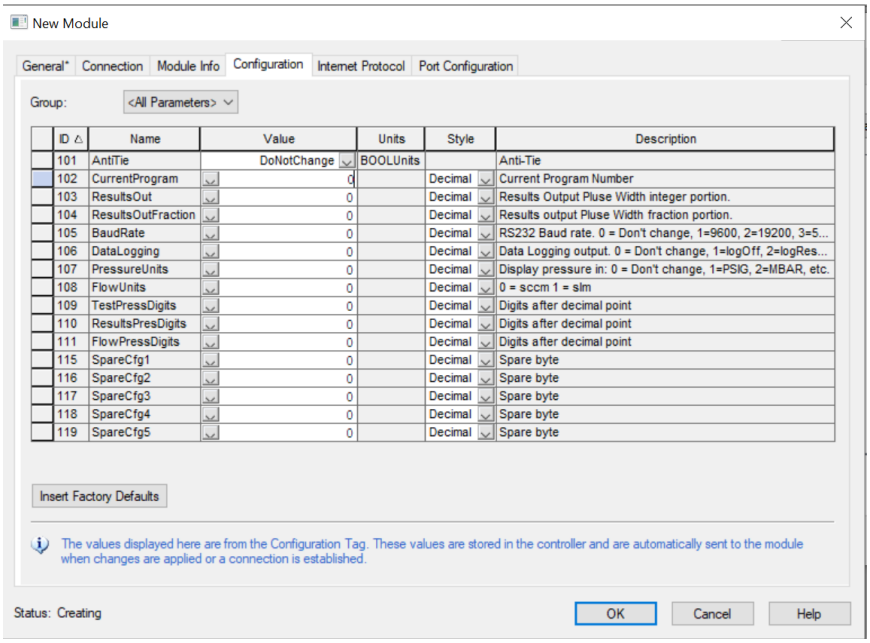


Figure 3 – Connection Type

6. The Connection tab allows the RPI value to be changed, the default value of 1000.0 milliseconds (e.g. 1 second) is the recommended value.



7. The Configuration tab allows the modification of the configuration parameters. The Configuration parameters are sent to the Isaac whenever the PLC creates an I/O type connection to the Isaac. The definition of each configuration parameters is provided below.



8. Clicking the **OK** button will finish the creation of the module, causing the newly created item to appear, as shown below in Figure 4 below.

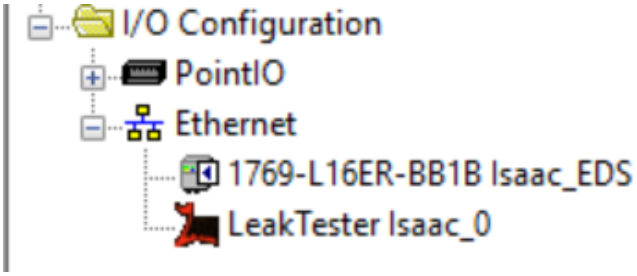


Figure 4 - Zaxis Isaac tester Ethernet I/P Module

Other module configuration items can be completed by double clicking on the newly created module and setting the desired parameters etc.

Generic Ethernet Module configuration

Configuring the Allen-Bradley 1769 CompactLogix Controller for communications with the Isaac tester via EtherNet/IP™ consists of creating a new *Generic Ethernet Module*. Use the following steps.

1. Start Logix Designer and Ensure the Communications are offline.
2. Create a new ETHERNET-MODLE by right clicking on the **Ethernet** menu item displayed on the **Controller Organizer** window (Figure 5) and select **New Module…** which should display the **Select Module Type** window (Figure 6).

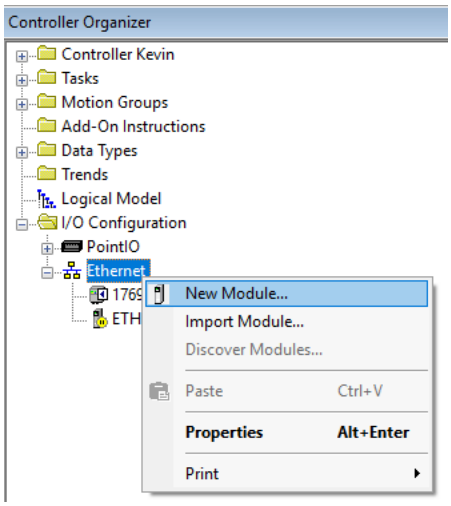


Figure 5 - Create New Ethernet-Module

3. With the **Select Module Type** window use the filter text box to find a **Generic Ethernet Module** as show below. Then click the **Create** button.

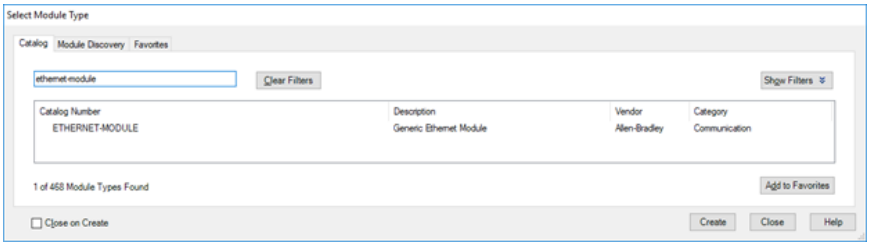


Figure 6 - Select Generic Ethernet Module

4. After clicking on the **Create** button the **New Module** information can be filled in.
    a. Enter a valid name
    b. Enter a description, if desired.
    c. Ensure the **Comm Format** is set to **Data - DINT**
    d. Enter the **IP address** of the Isaac tester (default I/P is: 192.168.2.130).
    e. Enter **1** for the **Input** Assembly Instance with a Size of **16**.
    f. Enter **2** for the **Output** Assembly Instance with a Size of **4**.
    g. Enter **4** for the **Configuration** with a Size of **16** (or 0 if Configuration data is not desired).

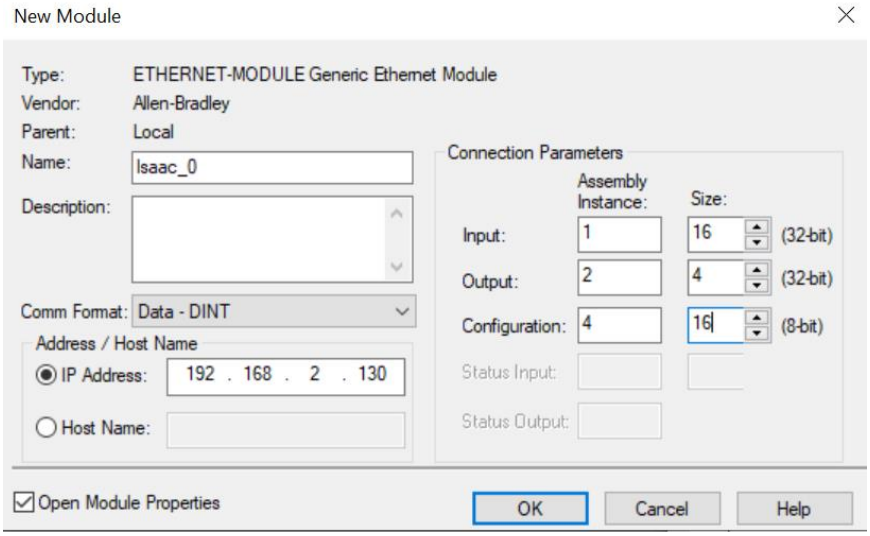


Figure 7 - New Ethernet module properties

The **Configuration** size is optional, it can be 0 (zero) or 16. When zero no initial configuration will be done when communications to the Isaac tester is established. When size is set to 16 a block of 16 bytes will be sent to the tester, which are used to do initial configuration. See Configuration Tags below.

5. Clicking the **OK** button creates the new module as well as the necessary controller tags used to interface with the Isaac tester and displays the **Connection** tab of the newly created module as seen in Figure 8 below. Set the **Requested Packet Interval (RPI)** to 40 e.g. 40 milliseconds. The

RPI should be adjusted after a significant number of test runs have been completed and a more accurate value can be determined.

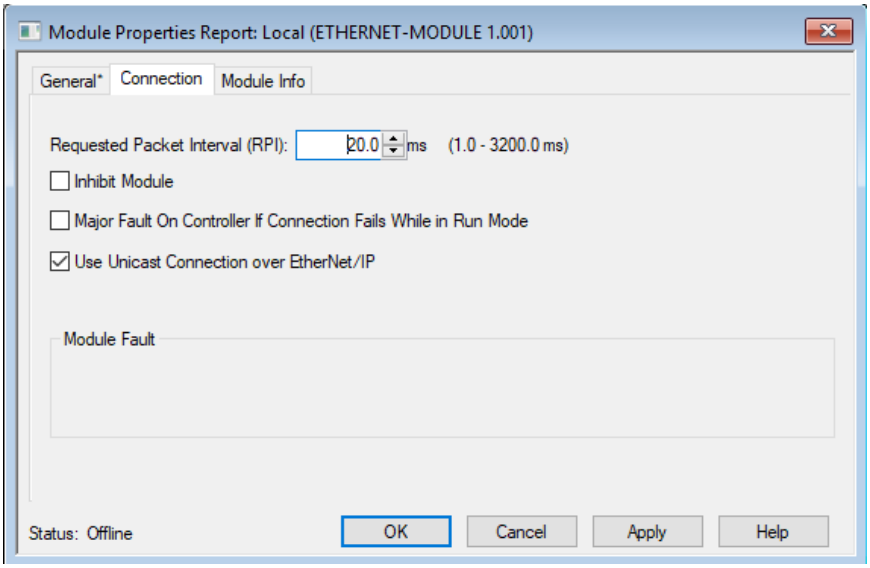


Figure 8 - Setting the RPI

6. Click the **Apply** button to save the properties of the newly created module.
7. Currently the Module Info tab is not functioning. The Zaxis vendor information must be registered and published by OVDA before he Module Info tab provides any meaningful information.
8. Open the **Controller Tags** to display the newly created tags for the Isaac module as shown below.

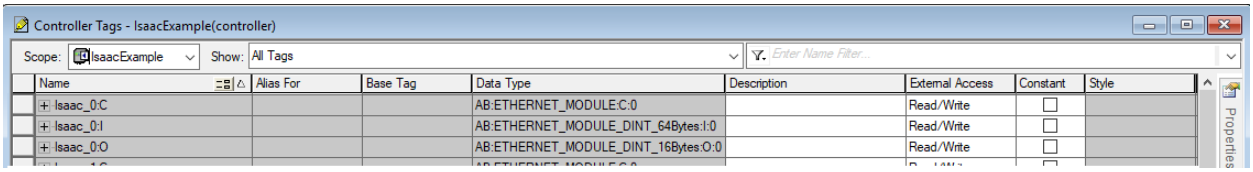


Figure 9 – Isaac controller tags

9. As can be seen in Figure 9 three new tag groups have been created which correspond to the Assembly Instances seen in Figure 7. Isaac_0:C corresponds to the Configuration assembly instance, if the size of the Configuration instance is zero the Isaac_0:C tag group will not exist. Tag group Isaac_0:I corresponds to the Input instance and Isaac_0:O to the Output instance.

Isaac Tag Groups

The tag groups for the Isaac tester will always have the Input and Output tags, the optional Configuration tags will only be available when a size other than zero is placed in the **Size** field of the Configuration Instance. The following tables: Table 3, Table 4 and Table 5, represent the structure of the different tag groups.

These tag groups have been defined in a Microsoft Excel spread sheet (Isaac_Udts.xlsx) available from Zaxis support and/or the Zaxis download web page.

Configuration Tags

The configuration tags are used to initialize selected parameters within the Isaac tester. These tags are not required and will not be sent to the VMP if the size of the Configuration instance is set to zero. If the Configuration size is non-zero, it must be set to the size indicated in step 4.g above, or undesirable results will occur.

Configuration tags are sent to the Isaac tester when the communications channel is opened, e.g. when the Ethernet module comes online.

Name	Data Type	Valid Values	Description
AntiTie	SINT	0,1,2,4	Test start when: 1 = input 1, 2 = input 1 & 2, 4 = input 1 & start button
CurrentProgram	SINT	0 thru 99	Starting program number
ResultsOut	SINT	> 0	Integer value of results out pulse width
ResultsOutFraction	SINT	> 0	Fraction of results out pulse width
BaudRate	SINT	1 thru 4	Baud rate for RS232 port. 1 = 9600, 2=19200, 3=57600, 4=115200
DataLogging	SINT	1 thru 4	When to log data on RS232 port. 1=Off, 2=Results, 3=0.1 sec, 4=1 sec
PressureUnits	SINT	1 thru 7	Units for pressure 1=PSIG, 2=MBAR, 3=MMHG, 4=INH2O, 5=KPA, 6=CMH20, 7=INHG
FlowUnits	SINT	0 or 1	Units for flow. 0=sccm, 1=slm
TestPresDigits	SINT	0 thru 3	Number of digits after decimal. 0 thru 3
ResultsPresDigits	SINT	0 thru 5	Number of digits after decimal. 0 thru 5
FlowPressureDigits	SINT	0 thru 3	Number of digits after decimal. 0 thru 3
SpareCfg1	SINT	0	Spare data
cosType	INT	0 to 0xFF	Indicates which change-of-states events should be reported
SpareCfg2	SINT	0	Spare data
SpareCfg3	SINT	0	Spare data

Table 3 – Ethernet-Module Configuration Tags

The TSi (touch screen interface) has input fields with the same or similar names to the configuration tags, more detailed information as to the purpose of these tags may be found in the Isaac User's Manual. These configuration tags preform the same functions as the TSi fields.

It is possible to set, or write, a subset of tags. When an invalid value is given, such as a -1, the given tag parameter will be ignored. For example, if a -1 is given for the *Program* the current program will not be changed.

cosType – Change-Of_State flags

The change-of-state flags indicate which events should be reported by the producer (the Isaac tester) to the consumer (usually the PLC). When the flag is set an Input message (e.g. a message is sent from the Isaac tester to the PLC) indicating which event caused the message see (CosStatus field in the PLC Input Tags section). The default is to only report the test results, which means when the EDS file is used to create the Isaac Ethernet modules and the COS connection is selected the only messages sent by the Isaac will be current status message (which will occur every RPI milliseconds) and the test results message. Thus ensuring that the PLC is sent a "heart beat" e.g. a status message every RPI milliseconds and the test results at the end of each test.

The change-of-state bits indicate the reason the message was produced by the Isaac tester.

1. CosStatus – Is set when a status message has been requested by the originator (e.g. the PLC) see the StatusGet bit in PLC Output Tags, or when the RPI has expired.
2. CosSequencePts – Is set when a test is running and the sequence has changed and/or the elapse time has changed.
3. CosTestStart – Is set at the start/beginning of a test.
4. CosTestsResults – Is set when a test has completed.
5. CosPressure – Is set when a test is not running but the pressure changes.
6. CosProgram – Is set when a program parameter has changed.
7. CosFirmwareCfg – Is set when a firmware parameter changed.
8. CosProgNumb – Is set then the current running program number is changed.
9. CosTestValve – Is set when adjusting the pressure and a valve is open/closed.

It is Zaxis recommendation that only the CosTestResults bit be set for normal operation. Setting the other bits may cause communication delays and communication disconnects.

PLC Output Tags

Output tags are sent by the PLC to the Isaac tester every **Request Packet Interval (RPI)** milliseconds as configured above, see Figure 8 - Setting the RPI. Once the Isaac scans i.e. process, the output tags from the PLC, the input tags are sent from the Isaac tester to the PLC.

Name	Data Type	Valid Values	Description
Start	BOOL	0 or 1	Bit 0 = Start the tests
StartAntiTie	BOOL	0 or 1	Bit 1 = Start on input 1 and 2 (Anti-Tie)
Abort	BOOL	0 or 1	Bit 2 = Stop the current test
Retest	BOOL	0 or 1	Bit 3 = Rerun the failed test
Valve1On	BOOL	0 or 1	Bit 4 = Turn pressure valve one on
Valve2On	BOOL	0 or 1	Bit 5 = Turn pressure valve two on
Valve3On	BOOL	0 or 1	Bit 6 = Turn pressure valve three on
Valve4On	BOOL	0 or 1	Bit 7 = Turn pressure valve four on
IgnoreCmds	BOOL	0 or 1	Bit 8 = Ignore EthierNet I/P commands
StatusGet	BOOL	0 or 1	Bit 9 = Get current device status
SpareBit10	BOOL	0	Spare bits 10 thru 31
CurrentProgram	SINT	-1 thru 99	Set to current program. -1 means don't change current program.
SpareByteOut1	SINT		Spare data
SpareIntOut1	INT		
SpareDIntOut1	DINT		
SpareDIntOut2	DINT		

Table 4 - Output Tags

Again, the tag names are similar or the same as the TSi field names, or buttons, e.g. Start is the green push button, Abort the red push button. The IgnoreCmds bit tells the Isaac tester to ignore any commands via EtherNet I/P. This bit should be set when TSi is being used to configure the Isaac tester to ensure there is no conflict between the PLC and the TSi.

The CurrentProgram tag allows the currently running program to be quickly changed to a different program. This could be used if the product being tested need to have a pressure decay test as well as a flow test. Program zero could be configured to run a pressure decay test and program one a flow test. The PLC could set the CurrentProgram to zero and start the test (set Start) if the product passed the pressure decay test, the PLC could change the CurrentProgram to one and start the test (set Start), which would then run the flow test.

PLC Input Tags

Input tags are sent to the PLC from the Isaac tester after the Output tags have been scanned, or processed, by the Isaac tester. Input tags provide the current state of the Isaac to the PLC. The following table list the tags returned from the Isaac tester.

Name	Data Type	Valid Values	Description
CosStatus	BOOL	0 or 1	Bit 0 = Status Update. E.g. heart beat
CosSequencePts	BOOL	0 or 1	Bit 1 = Test Pressure and/or time
CosTestStart	BOOL	0 or 1	Bit 2 = Test Started
CosTestResults	BOOL	0 or 1	Bit 3 = Test Completed
CosPressure	BOOL	0 or 1	Bit 4 = Pressure changed

CosProgram	BOOL	0 or 1	Bit 5 = Program data changed
CosFirmwareCfg	BOOL	0 or 1	Bit 6 = Firmware configuration changed
CosProgNumb	BOOL	0 or 1	Bit 7 = Current running program number changed
CosTestValve	BOOL	0 or 1	Bit 8 = Test valve changed
ClassId	INT	0 or 1	Id of the class that changed state
Instance	INT	1 thru 99	Instance Id of the class that changed state
Attribute	INT	1 thru 99	Attribute number that changed state
Busy	BOOL	0 or 1	Bit 0 = Isaac is busy
GlobalPass	BOOL	0 or 1	Bit 1 = All the ports passed the test
GlobalFail	BOOL	0 or 1	Bit 2 = One or more ports failed the test
Fail_1	BOOL	0 or 1	Bit 3 = Port 1 failed
Fail_2	BOOL	0 or 1	Bit 4 = Port 2 failed
Fail_3	BOOL	0 or 1	Bit 5 = Port 3 failed
Fail_4	BOOL	0 or 1	Bit 6 = Port 4 failed
Pass_1	BOOL	0 or 1	Bit 7 = Port 1 Passed
Pass_2	BOOL	0 or 1	Bit 8 = Port 2 Passed
Pass_3	BOOL	0 or 1	Bit 9 = Port 3 Passed
Pass_4	BOOL	0 or 1	Bit 10 = Port 4 Passed
AntiTie	BOOL	0 or 1	Bit 11 = Anti Tie is enabled
Interlock	BOOL	0 or 1	Bit 12 = Interlock is required
Abort	BOOL	0 or 1	Bit 13 = Test Aborted
SpareBit14	BOOL	0 or 1	Bit 14 = Spare Bit
Fault	BOOL	0 or 1	Bit 15 = Faulted
SpareByteIn1	SINT	0	Spare data
TestPresValve1	BOOL	0 or 1	Bit 0 = State of Test Pressure Vavle 1
TestPresValve2	BOOL	0 or 1	Bit 1 = State of Test Pressure Vavle 2
TestPresValve3	BOOL	0 or 1	Bit 2 = State of Test Pressure Vavle 3
TestPresValve4	BOOL	0 or 1	Bit 3 = State of Test Pressure Vavle 4
CurrentProgram	SINT	0 thru 99	Current program
TestType	SINT	0 thru 11	Type of test being run. PD, POCC, VD, VOCC, Flow, etc.
PortEnabled1	BOOL	0 or 1	Bit 0 = Port 1 enabled
PortEnabled2	BOOL	0 or 1	Bit 1 = Port 2 enabled
PortEnabled3	BOOL	0 or 1	Bit 2 = Port 3 enabled
PortEnabled4	BOOL	0 or 1	Bit 3 = Port 4 enabled
Sequence	SINT	0 thru 9	Current test sequence 0 = Idle.
SpareIntIn1	INT	0	Spare Int
ElapseTime	DINT	> 0	Sequence elapse time in 1/10's seconds.
PortStatus1	SINT	0 thru 17, 250 thru 255	Status of Port 1. E.g. Testing, Leak, Occl, Perr
PortStatus2	SINT		Status of Port 2. E.g. Testing, Leak, Occl, Perr
PortStatus3	SINT		Status of Port 3. E.g. Testing, Leak, Occl, Perr
PortStatus4	SINT		Status of Port 4. E.g. Testing, Leak, Occl, Perr
PressurePort1	DINT		Test pressure * 1000. E.g. fixed-point decimal
PressurePort2	DINT		Test pressure * 1000. E.g. fixed-point decimal
PressurePort3	DINT		Test pressure * 1000. E.g. fixed-point decimal
PressurePort4	DINT		Test pressure * 1000. E.g. fixed-point decimal

FlowPort1	DINT		Flow pressure * 1000. E.g. fixed-point decimal
FlowPort2	DINT		Flow pressure * 1000. E.g. fixed-point decimal
FlowPort3	DINT		Flow pressure * 1000. E.g. fixed-point decimal
FlowPort4	DINT		Flow pressure * 1000. E.g. fixed-point decimal
SpareDIntIn1	DINT		

Table 5 - Input Tags

COS Change-of-State bits

Change-of-state messages are produced only when the originator (e.g. usually the PLC) sets the corresponding bit in the configuration tags.

The Busy bit preforms the same functions as the Busy input signals as described in the User's Manual, e.g. the tester is busy preforming a test.

The other bits are described in the Isaac User's manual.

Isaac Firmware/Program tags

The Isaac has two different types of tags: 1) tags that remain the same regardless of which program is running and 2) tags for each individual program. This document will refer to these different types as firmware paraments and program parameters respectively.

Regardless of the type of parameter (firmware or program) both are saved in flash memory and are read into RAM (random access memory) each time the Isaac is powered up and/or reset. The flash memory doesn't require a battery and is therefore more reliable for longer periods of time.

This also implies that these parameters do not (and should not) be written to the Isaac tester each time a program (test) is run. Doing so causes unnecessary overhead and slows the Isaac's response times which may cause timeout problems.

Firmware Parameters

Firmware parameters are those parameters that remain the same regardless of which program (or test) is being executed. Firmware parameters are typically set once on initial setup of the Isaac tester and very seldom changed. The following table describes the Firmware parameters.

Name	Data Type	Valid Values	Description
CosType	INT	Read Only	Change-of-State flags
ClassId	INT	Read Only	Id of the class that changed state
Instance	INT	Read Only	Instance Id of the class that changed state
Attribute	INT	Read Only	Attribute number that changed state
LockPrgEdits	BOOL	0 or 1	Tsi Program parameter editing is locked
LockCurrentPrgEdits	BOOL	0 or 1	Tsi Current program editing is locked
LockCalibrationEdits	BOOL	0 or 1	Tsi Calibration edits are locked
ElectronicReg	BOOL	0 or 1	Electronic regulator installed, else Manual Reg
Concurrent	BOOL	0 or 1	Run test concurrently on all channels

BuzzerEnabled	BOOL	0 or 1	Buzzer is enabled
BusyImm	BOOL	0 or 1	0 = send busy msessage after all overhead, 1 = send busy with pass/fail
DhcpEnabled	BOOL	0 or 1	DHCP is enabled
OkToResetDevice	BOOL	0 or 1	Ok to reset the device because of date/time changes etc.
SpareBit9	BOOL	0	Spare data
SpareBit10	BOOL	0	Spare data
SpareBit11	BOOL	0	Spare data
SpareBit12	BOOL	0	Spare data
SpareBit13	BOOL	0	Spare data
SpareBit14	BOOL	0	Spare data
SpareBit15	BOOL	0	Spare data
TesterNumber	SINT	1 to 128	Device number on 485 bus
BaudRate	SINT	0 thru 4	0 = Don't change, 1=9600, 2=19200, 3=57600, 4=1152
DataLogging	SINT	0 thru 4	0 = Don't change, 1=LogOff, 2=logResult, 3=log 0.1 secs, 4= log 1.0 sec
PressureUnits	SINT	0 thru 7	0 = Don't change, 1=PSIG, 2=MBAR, 3=MMHG,4=INH2O, 5=KPA, 6=CMH2O, 7=INHG
FlowUnits	SINT	0 or 1	0 = sccm 1 = slm
TestPresDigits	SINT		Number of digits after test pressures 0 to 3
ResultsPresDigits	SINT	0 thru 5	Number of digits after results pressures 0 to 5
FlowPresDigits	SINT	0 thru 3	Number of digits after flow pressures 0 to 3
BCD	SINT	0 thru 99	BCD Inputs. See documentation
NumberChannels	SINT	1 thru 4	Number of installed channels/ports
Sensors	SINT	1 thru 4	Number of installed sensors. 1st nibble = Pressures sensors, 2nd nibble = Flow.
AntiTie	SINT	1, 2, or 4	Test start when: 1 = input 1, 2 = input 1 & 2, 4 = input 1 & start button
MachineType	SINT	O thru 10	Types of tests that can be preformed
SpareByte1	SINT	0	Spare data
IoResetTime	INT	0 thru 3000	Number of milliseconds the I/O remains in a given state
SpareByte2	SINT	0	Spare data
SpareByte3	SINT	0	Spare data
SerialNumber	STRING_16	Read Only	Serial number. 16 bytes.
IpAddress	STRING_16	Valid I/P	I/P Address. 16 bytes.
IpNetmask	STRING_16	Valid I/P	I/P Netmask. 16 bytes.
IpGateway	STRING_16	Valid I/P	I/P Gateway. 16 bytes.
MacAddress	STRING_20	Read Only	MAC Address. 20 bytes.
DateTime	STRING_20	Valid Date/Time	Date and time: YYYY/MM/DD hh:mm:ss
ModelNumber	STRING_24	Read Only	Model number or TIN
OptionsAndFittings	STRING_32	Read Only	Installed options and fittings

SccsRevDate	STRING_24	Read Only	Source Code Control Date stamp
SccsRev	STRING_12	Read Only	Source Code Revision
SccsRevRange	STRING_24	Read Only	Source Coe Revision range

Table 6 - Configuration Parameters

The above configuration parameters are typically defined as a User-Defined Type (UDT) an imported from a Microsoft Excel spread sheet (Isaac_Utds.xlsx) which can be downloaded from the Zaxis web site.

Reading and Writing of Firmware Parameters

Firmware parameters can be read and written using the PLC Message (MSG) instruction.

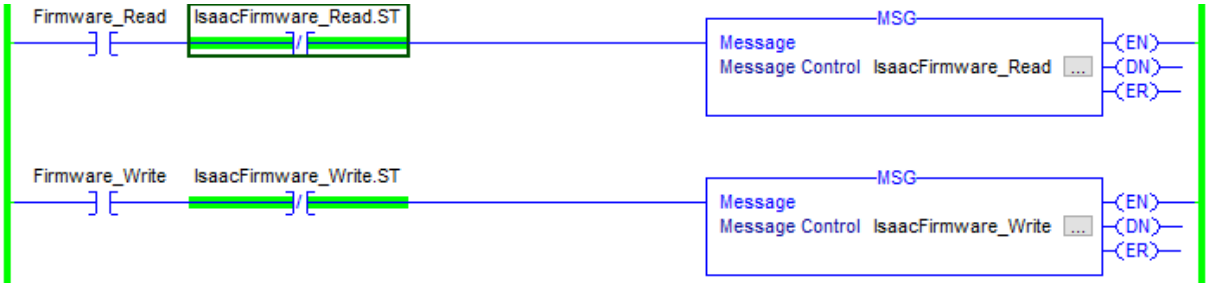


Figure 10 - MSG Tag

Firmware parameters are read using the CIP Read All Attributes message and written using the CIP Write All Attributes message. These CIP messages (Read/Write All Attributes) are created on the PLC using the *CIP Generic* **Message Type** with a **Service Code** of 1 (e.g. Read all attributes) or 2 (e.g. Write all attributes), a vendor specific **Class** code of 0x64 (Hex) and an **Instance** of 1. As show below in **Error! R**

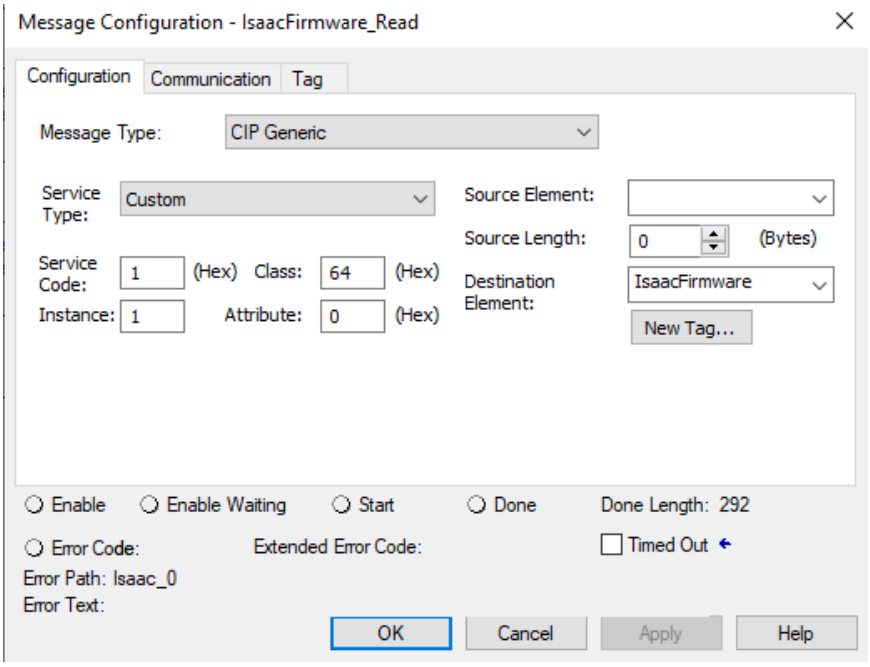


Figure 11 - Read Isaac Configuration MSG

eference source not found. and Figure 14 below.

The **Destination Element** should be a *User-Defined* data type which minors the tags described in Table 6.

The *Communication* tab references the **Path** of the VMP Ethernet module that was added above in the Isaac Tester Configuration for Ethernet/IP™ section, as show below in Figure 12.

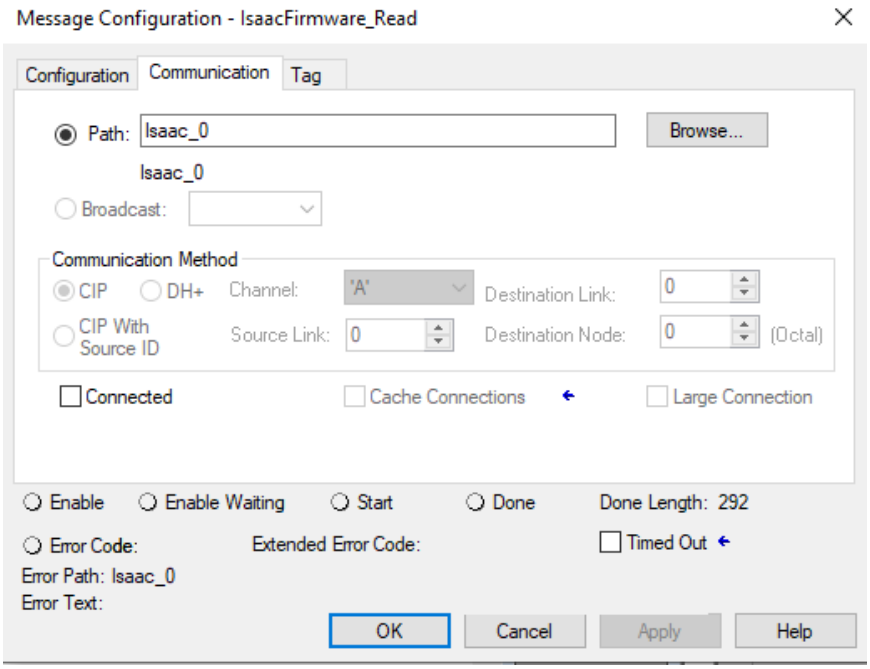


Figure 12 - MSG Communication tab

The MSG Tag can be given any meaningful name as shown in Figure 13 below to complete the configuration.

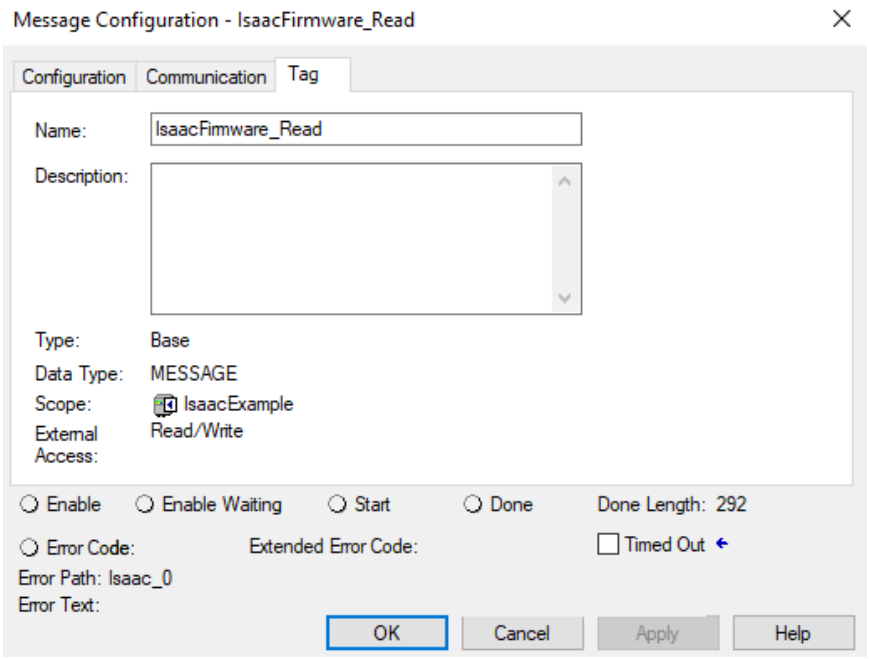


Figure 13 - MSG Tag tab

The only difference between reading and writing setup parameters is the **Service Code**, it changes from 1 to a 2, and the **Source Element** now references the user defined data to be written. The MSG parameters for writing are show in Figure 14 below.

The *Communication* tab and *Tag* tab have the same parameters as reading the setup.

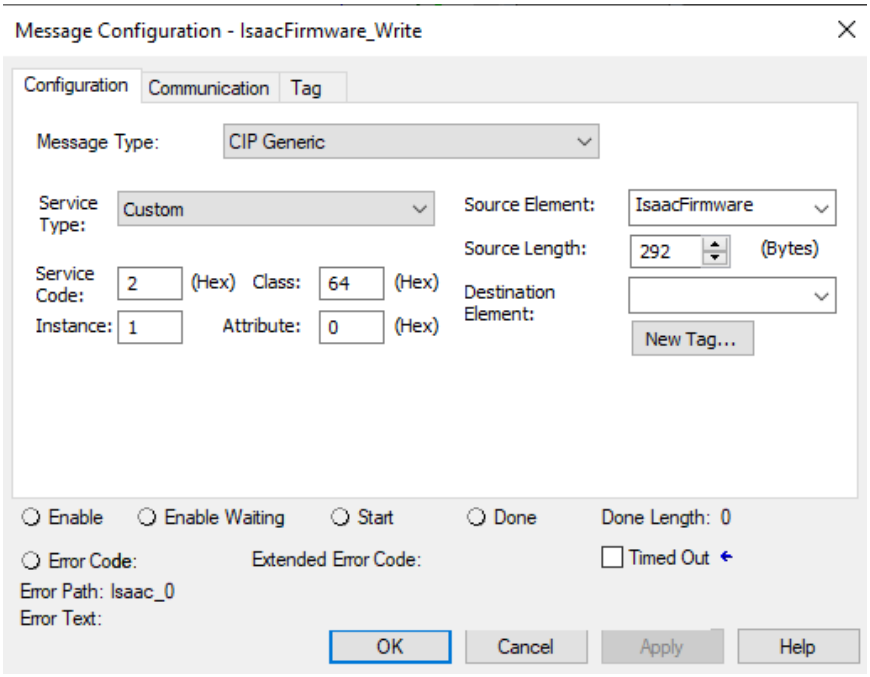


Figure 14 - Write Isaac Firmware Configuration MSG

Program Parameters

Program parameters allow the VMP to be configured such that different programs or loops can be run by the same VMP yet perform completely different tasks. Programs can be linked allowing one program to execute after the parent program, with each program having different parameters. Table 7 shows the different program parameters and their meaning.

Name	Data Type	Valid Values	Description
CosType	INT	Read Only	Change-of-State flags
ClassId	INT	Read Only	Id of the class that changed state
Instance	INT	Read Only	Instance Id of the class that changed state
Attribute	INT	Read Only	Attribute number that changed state
PrgNumb	SINT	0 thru 99	Program number
SpareByte1	SINT	0	Spare Byte 1
SpareIntIn1	INT	0	Spare integer
PortEnabled1	BOOL	0 or 1	Port 1 is enabled
PortEnabled2	BOOL	0 or 1	Port 2 is enabled
PortEnabled3	BOOL	0 or 1	Port 3 is enabled
PortEnabled4	BOOL	0 or 1	Port 4 is enabled
FixtureValve1	BOOL	0 or 1	Keep Fixture valve 1 on during test step
FixtureValve2	BOOL	0 or 1	Keep Fixture valve 2 on during test step
ClampHoldFail	BOOL	0 or 1	Hold clamps on fail
Rslts232	BOOL	0 or 1	Output results to RS232 port
RsltsEthr	BOOL	0 or 1	Output results to Ethernet port 23

AutoFFill	BOOL	0 or 1	Auto Fast Fill or timed
LeakStd	BOOL	0 or 1	Leak Standard
AutoVent	BOOL	0 or 1	Auto Vent or timed
IncreaseLimit	BOOL	0 or 1	Enable Increase Limit
EvalAtEnd	BOOL	0 or 1	Evaluate at End of Test
SpareBit_14	BOOL	0 or 1	Spare Data bits 14 thru 31
TestType	SINT	0 thru 10	Type of test to run. PD, POCC, VD, VOCC, Flow, Burst, Vburst, Crack, Creep, Flow2, Reseat
NextPrg	SINT	-1 thru 99	Next program to run. -1 means not linked.
ElectRegDelay	SINT	0 thru 100	% of fill timer to allow for Electronic Regulator to seek
SpareByte2	SINT	0	Spare
TimeIdle	DINT	0	Reserved. Do not use.
TimeClmp1	DINT	0 thru 10000	Elapse time in 1/10 seconds for Clamp 1 Sequence
TimeClmp2	DINT	0 thru 10000	Elapse time in 1/10 seconds for Clamp 2 Sequence
TimeFFill	DINT	0 thru 10000	Elapse time in 1/10 seconds for Fast Fill (Pre-Fill) Sequence
TimeFill	DINT	0 thru 10000	Elapse time in 1/10 seconds for Fill Sequence
TimeSettle	DINT	0 thru 10000	Elapse time in 1/10 seconds for Settle Sequence
TimeTest	DINT	0 thru 10000	Elapse time in 1/10 seconds for Test Sequence
TimeVent	DINT	0 thru 10000	Elapse time in 1/10 seconds for Vent Sequence
TimeUnClmp1	DINT	0 thru 10000	Elapse time in 1/10 seconds for Unclamp 2 Sequence
TimeUnClmp2	DINT	0 thru 10000	Elapse time in 1/10 seconds for unclamp 1 Sequence
SpareDInt1	DINT	0	Spare
SpareDInt2	DINT	0	Spare
TestPres	DINT		Test Pressure
TestPresMin	DINT		Minimum Test Pressure
TestPresMax	DINT		Maximum Test Pressure
RegKp	DINT	0 thru 10000	Regulator P-I Kp
RegKi	DINT	0 thru 10000	Regulator P-I Ki
RegKd	DINT	0 thru 10000	Regulator PI Kd
PresFFill	DINT		Pre Fill Pressure
PresTestMin	DINT	0 thru 10000	Minimum test pressure. E.g. Decay, Min Crack/Burst/Flow etc.
PresTestMax	DINT	0 thru 10000	Maximum test pressure e.g. Increase, Max Crack/Burst/Flow etx.
PresTestTrigger	DINT	0 thru 10000	Trigger test pressure e.g. Trigger Crack/Burst etc.

Volume	DINT	0 thru 10000	Air volume used to calculate approx. Leak rate. 0 to disable calculation.
RampRate	DINT	0 thru 100	How fast to ramp up the the Fast Fill pressure
SpareDInt3	DINT	0	Spare
PrgName	STRING_16	Printable Chars	Program name length (4 bytes) + Program name (16 bytes)

Table 7 - Program Configuration Parameters

As with the configuration parameters, the program parameters should be used to define a **User-Define** data type.

Program parameters are read using the CIP Read All Attributes message and written using the CIP Write All Attributes message. These CIP messages (Read/Write All Attributes) are configured on the PLC using the *CIP Generic* **Message Type** with a **Service Code** of 1 (e.g. Read all attributes) or 2 (e.g. Write all attributes), a vendor specific **Class** code of 0x65 (Hex) and an **Instance** of 1. As show below in Figure 16, Figure 17 and Figure 17.

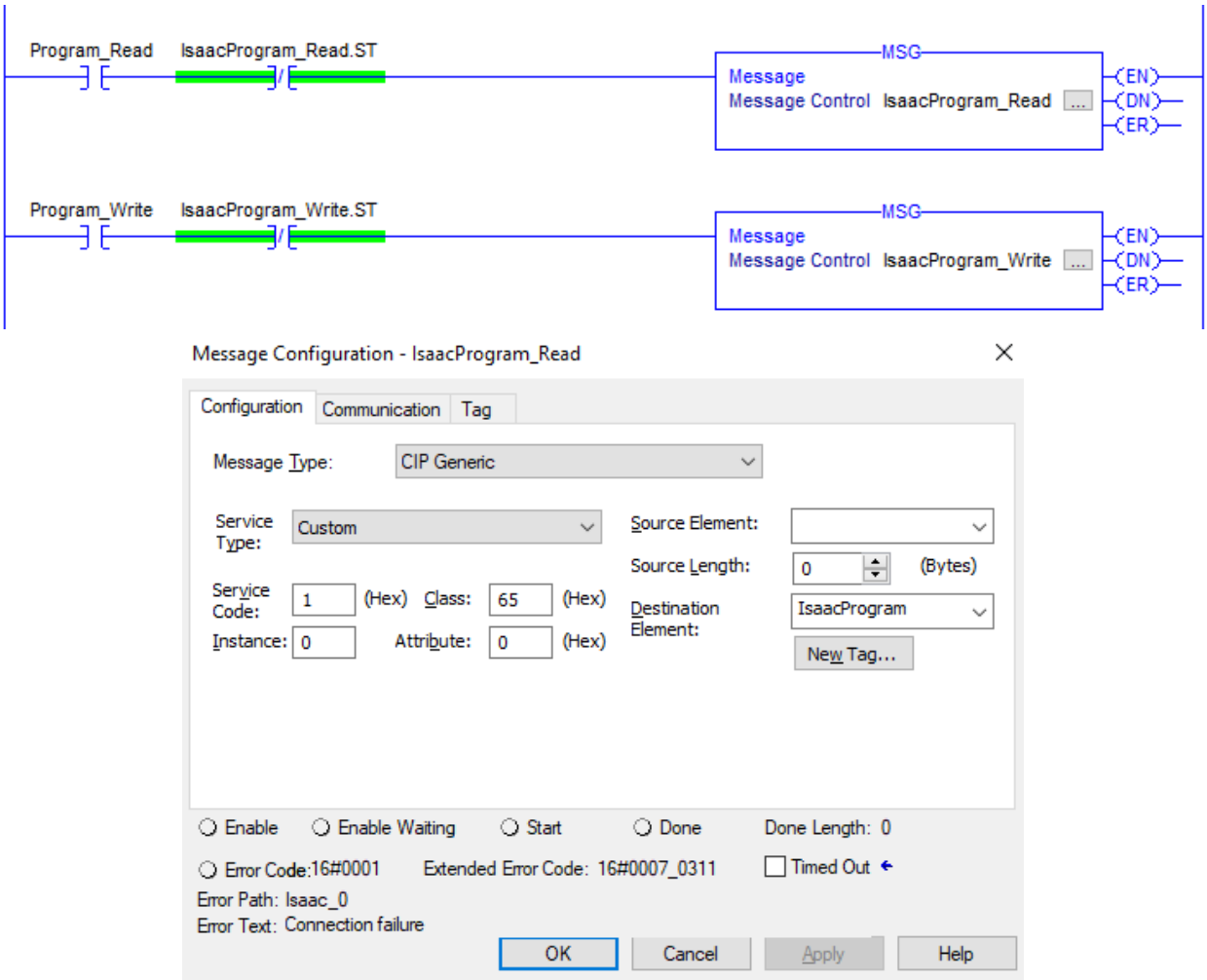


Figure 15 - Reading/Writing program Ladder Logic

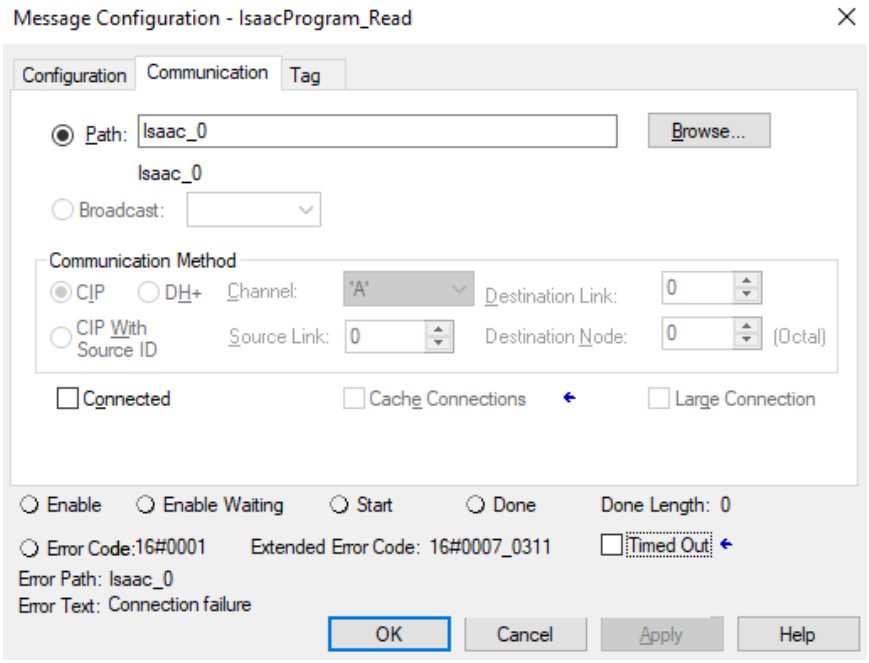


Figure 16 – Read Isaac Program Parameters MSG

Writing Program Parameters is the same as reading with the acceptation of the **Service Code** which should be changed to 2 e.g. Write All Attributes, as show in Figure 14 below.

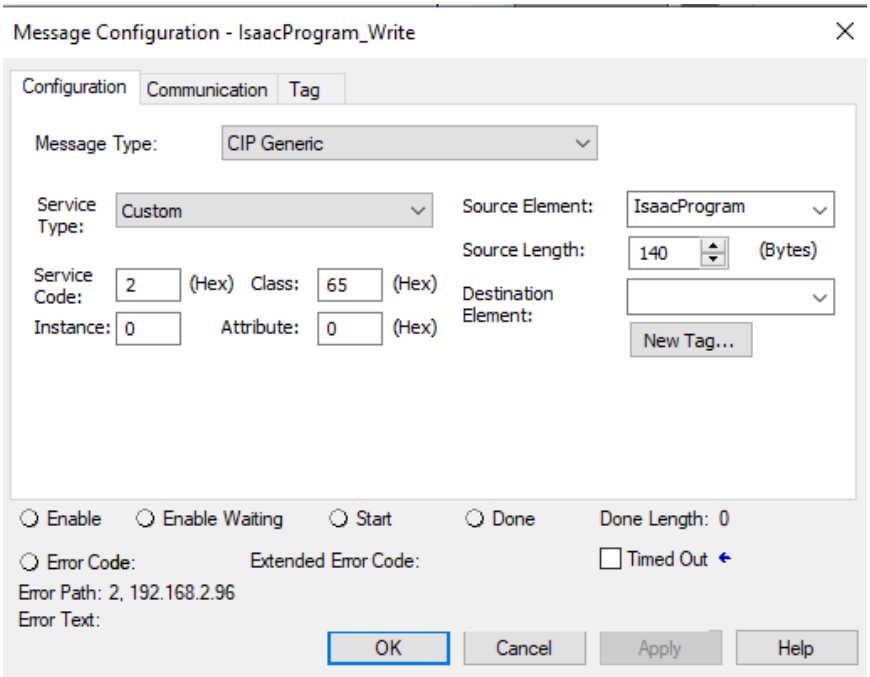


Figure 17 - Write VMP Program Parameters MSG

Multiple Isaac testes can be configure using the same MSG instruction following the examples shown in the "Logix 5000 Controllers Message Programming Manual" created by Rockwell automation. See https://literature.rockwellautomation.com/idc/groups/literature/documents/pm/1756-pm012_-en-p.pdf Chapter 3 for more information.

TCP/IP Network Configuration

The default TCP/IP settings are provided in Table 2 above, however these values can be reconfigured using the predefined CIP TCP/IP object. The object definition is shown in Table 8, and is based on the *TCP/IP Interface Object* in *Volume 2: EtherNet/IP Adaptation of CIP, Chapter 5: Object Library*.

Name	Data Type	Valid Values	Description
Status	DINT	Read only	Interface Status bits. 0x02=Configured from DHCP, 0x04=Hardware configuration.
Configuration Capability	DINT	Read only (0x34)	Bit map of interface capability flags: 0x04=DHCP capable 0x10=Interface configurable 0x20=Hardware configurable
Configuration Control	DINT		Interface control flags: 0x00=Use static configuration 0x02=Use DHCP configuration
Path Size	INT	Read only (3)	Number of 16-bit words in the following padded EPATH.
Path	SINT[4]	Read only	Padded EPATH to physical link object.
IP Address	DINT	0 to 0xFFFFFFFF	Devices I/P address.
Network Address	DINT	0 to 0xFFFFFFFF	Devices network address.
Gateway Address	DINT	0 to 0xFFFFFFFF	Devices gateway I/P address.
Name Server	DINT	0 to 0xFFFFFFFF	Primary name server I/P address.
Name Server 2	DINT	0 to 0xFFFFFFFF	Secondary name server I/P address.
Domain Name	STRING	ASCII Characters	Default domain name.
Host Name	STRING	ASCII Characters	Default host name.
Safety Network Number	SINT[6]	0 to 100	Ignored by VMP.
TTL	SINT	1 to 255	Time-to-Live value for IP Multicast packets. Ignored by VMP.
Alloc Control	SINT	0	Multicast address allocation. Ignored by VMP.
Reserved	SINT	0	Reserved for future use.
Num Mcast	INT	0	Number of multicast addresses to allocate. Ignored by VMP.
Mcast Start Addr	DINT	0	Starting multicast address. Ignored by VMP.
Selected Acd	BOOL	0	Activates use of ACD. Ignored by VMP.

Acd Active	SINT	0	State of ACD activity. Ignored by VMP.
Remote MAC	SINT[6]	0	MAC Address of remote node. Ignored by VMP.
Arp Pdu	SINT[28]	0	ARP PDU. Ignored by VMP.
EthernetIP Quick Connect	BOOL	0	Quick connect feature. Ignored by VMP.
Encp Inactivity Timeout	INT	0	Inactivity Timeout. Ignored by VMP.

Table 8 - TCP/IP Object definition

As can be seen from the table above many of the parameters are not required by the Isaac. However, this object does allow the reconfiguration of necessary TCP/IP parameters, e.g. I/P address, enable/disable DHCP etc.

TCP/IP parameters are read using the CIP Read All Attributes message and written using the CIP Write All Attributes message. These CIP messages (Read/Write All Attributes) are configured on the PLC using the *CIP Generic* **Message Type** with a **Service Code** of 1 (e.g. Read all attributes) or 2 (e.g. Write all attributes), a CIP **Class** code of 0xF5 (Hex) and an **Instance** of 1. As show in FIGURE 18 and Figure 19.

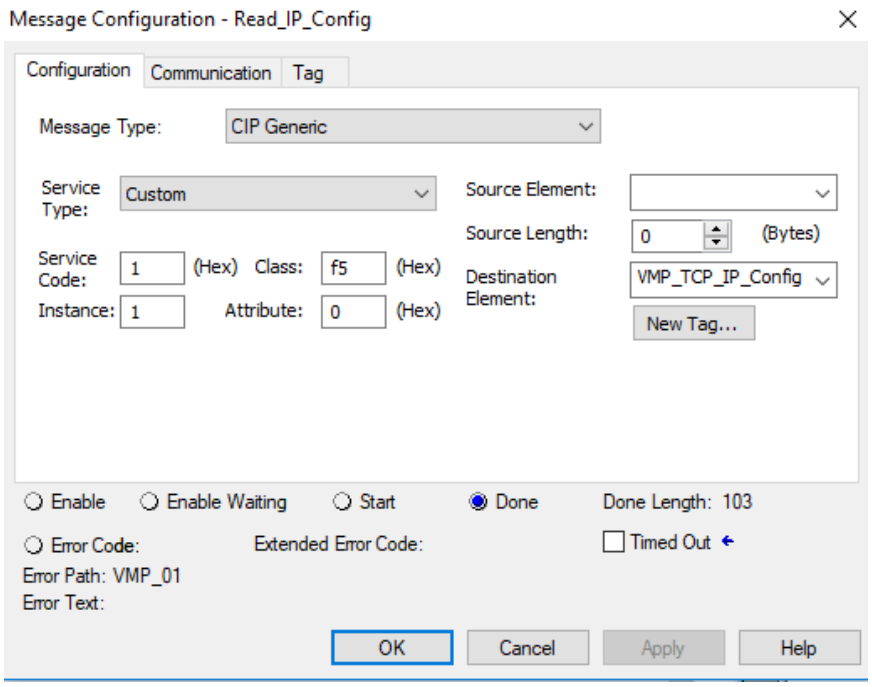


FIGURE 18 - Read TCP/IP Configuration Parameters

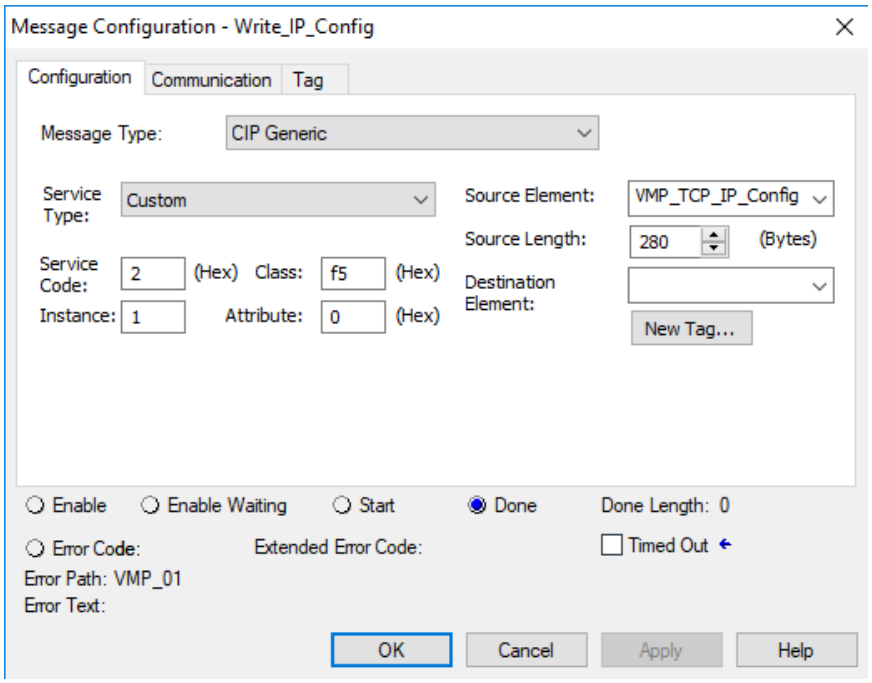


Figure 19 - Write TCP/IP Configuration Parameters

For example, to configure the VMP to use DHCP, simply set the Configuration Capability field to 0x02 and write the TCP/IP object using a MSG instruction.

To set anyone of the I/P addresses (e.g. I/P address, network address, gateway address, etc.) convert each address octet into hex and enter the hex number into the desired field. For example, to set the I/P address of the VMP to 192.168.168.100 enter the hex value 0xC0A8A864 into the IP Address field and write the TCP/IP object to the VMP.

Decimal:	192	168	168	100
Hex:	0xC0	0xA8	0xA8	0x64

Table 9 - I/P Address to Hex conversion example

The VMP default TCP/IP configuration can always be restored by pressing and holding the reset button for less than two seconds. If the button is held for longer than two seconds the VMP will use DHCP to configure TCP/IP.

Trouble Shooting

When problems occur, either in setting up the communications, or communicating with, the VMP, sometimes it is helpful to get more detailed than a simple fault error. The VMP displays error and status messages when a network connection has been established on port 8080. These diagnostic messages may give more detail and/or a better understanding of the problems.

A connection to the VMP diagnostics console can be created using the *Telnet* utility, available on all Unix based system and most Windows system. However, it may need to be turned on under windows. The following figure shows a check mark next to the **Telnet Client**, indicating that it is available for use.

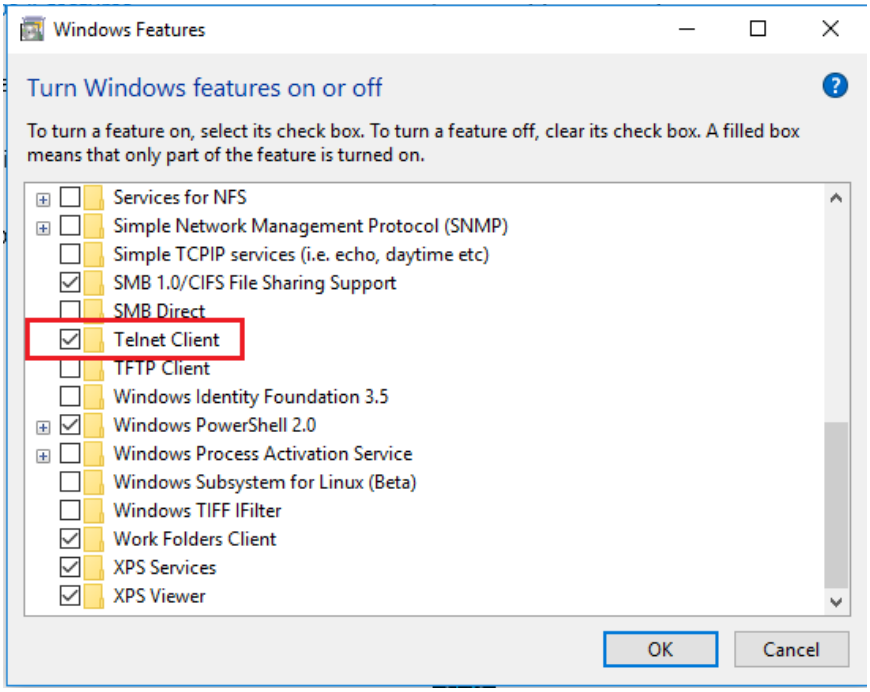


Figure 20 - Windows Telnet Client

With the *Telnet* utility available a connection can, be made using the following command, within a command window (which can be created by "*run cmd*"):

telnet <VMP_IP_address> 8080 e.g. *telnet 192.168.2.130 8080*

Figure 21 shows the output from the diagnostics console. With a line showing a buffer length mismatch, e.g. 18 32-bit words (72 bytes) should be 16 32-bit words (64 bytes). In this case the error was caused by using a size of 18 for the Input Assembly Instance when it should have been 16.

Other and more detailed information can be obtained from the diagnostics console; however, it is recommended that a Zaxis support personal help you in diagnosing and resolving more advanced problems.

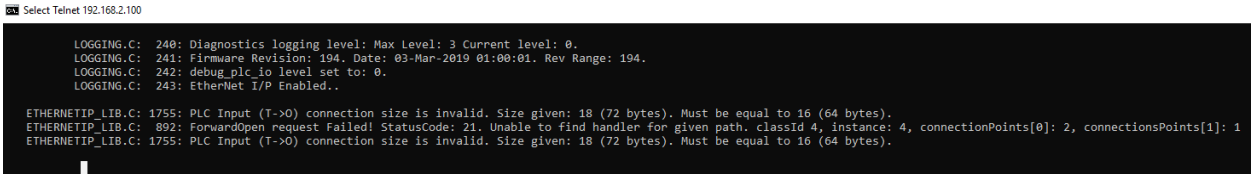


Figure 21 - Diagnostics Console